Installing on Windows

Download the Server

Description	Download	
Installer for Windows systems (from rabbitmq.com)	**rabbitmq-server-3.6.9.exe**	(Signature)
Installer for Windows systems (from github.com)	**rabbitmq-server-3.6.9.exe**	(Signature)

Uninstall previous version

If you have an existing installation and are planning to upgrade the Erlang VM from a 32bit to a 64bit version then you must uninstall the broker before upgrading the VM. The installer will not be able to stop or remove a service that was installed with an Erlang VM of a different architecture.

Install the Server

Firstly, download and run the **Erlang Windows Binary File**. It takes around 5 minutes.

Then just run the installer, `rabbitmq-server-3.6.9.exe`. It takes around 2 minutes, and will set RabbitMQ up and running as a service, with a default configuration.

Run RabbitMQ Service

Customise RabbitMQ Environment Variables
: The service will run fine using its default settings. You may want to **customise the RabbitMQ environment** or edit **configuration**.

Run RabbitMQ
: The RabbitMQ service starts automatically. You can stop/reinstall/start the RabbitMQ service from the Start Menu.

Manage the Service
: You can find links to RabbitMQ directories in the Start Menu.

: There is also a link to a command prompt window that will start in the sbin dir, in the Start Menu. This is the most convenient way to run the **various command line tools**.

Port Access

Firewalls and other security tools may prevent RabbitMQ from binding to a port. When that happens, RabbitMQ will fail to start. Make sure the following ports can be opened:

- 4369: **epmd**, a peer discovery service used by RabbitMQ nodes and CLI tools
- 5672, 5671: used by AMQP 0-9-1 and 1.0 clients without and with TLS
- 25672: used by Erlang distribution for inter-node and CLI tools communication and is allocated from a dynamic range (limited to a single port by default, computed as AMQP port + 20000). See **networking guide** for details.
- 15672: **HTTP API** clients and **rabbitmqadmin** (only if the **management plugin** is enabled)
- 61613, 61614: **STOMP clients** without and with TLS (only if the **STOMP plugin** is enabled)
- 1883, 8883: (**MQTT clients** without and with TLS, if the **MQTT plugin** is enabled
- 15674: STOMP-over-WebSockets clients (only if the **Web STOMP plugin** is enabled)
- 15675: MQTT-over-WebSockets clients (only if the **Web MQTT plugin** is enabled)

It is possible to **configure RabbitMQ** to use different ports.

Default user access

The broker creates a user `guest` with password `guest`. Unconfigured clients will in general use these credentials. By default, these credentials can only be used when connecting to the broker as localhost so you will need to take action before connecting from any other machine.

See the documentation on **access control** for information on how to create more users, delete the `guest` user, or allow remote access to the `guest` user.

Managing the Broker

To stop the broker or check its status, use `rabbitmqctl.bat` in `sbin` (as an administrator).

Stopping the Broker
: Use `rabbitmqctl stop`.

Checking the Broker Status

In This Section

Install: Windows
Install: Debian / Ubuntu
Install: RPM-based Linux
Install: Mac OS X
Install: Homebrew
Install: Windows (manual)
Install: Generic Unix
Install: Solaris
Install: EC2

Supported Platforms
Changelog
Erlang Versions
Signed Packages

Java Client Downloads
.NET Client Downloads
Erlang Client Downloads
Community Plugins

Nightly Builds

Related Links
Windows Quirks

Use `rabbitmqctl status`. All rabbitmqctl commands will report the node absence if no broker is running (i.e. nodedown).

More **info on rabbitmqctl**

Logging

Output from the server is sent to a RABBITMQ_NODENAME.log file in the RABBITMQ_LOG_BASE directory. Additional log data is written to RABBITMQ_NODENAME-sasl.log.

The broker always appends to the log files, so a complete log history is retained.

You can rotate logs using `rabbitmqctl rotate_logs`.

Troubleshooting When Running as a Service

In the event that the Erlang VM crashes whilst RabbitMQ is running as a service, rather than writing the crash dump to the current directory (which doesn't make sense for a service) it is written to an `erl_crash.dump` file in the base directory of the RabbitMQ server (set by the RABBITMQ_BASE environment variable, defaulting to `%APPDATA%\%RABBITMQ_SERVICENAME%` - typically `%APPDATA%\RabbitMQ` otherwise).

Windows-specific Issues

We aim to make RabbitMQ a first-class citizen on Windows. However, sometimes there are circumstances beyond our control. Please consult the **Windows-specific Issues** page.

Getting Help

If you have questions or need help, feel free to ask on **RabbitMQ mailing list**.